**Dado el siguiente código, y siendo la variable “con” un objeto *Connection*:**

***String sql = "UPDATE CLIENTE SET DOMICILIO=?, TLF=? WHERE***

***NIF=?;";***

***PreparedStatement sentencia;***

***int afectados;***

***sentencia = con.prepareStatement(sql);***

***afectados = sentencia.executeUpdate();***

**¿Qué falta para que funcione correctamente?**



No falta nada, es correcto



Falta asignar valores a los parámetros especificados por las interrogaciones, ejecutando métodos de tipo sentencia.setString(…).



Hay que corregir la sentencia SQL porque no es correcta, debemos sustituir las interrogaciones por valores reales.

Los objetos PreparedStatement definen una sentencia SQL con parámetros que son identificados por caracteres de interrogación y que deben ser sustituidos mediante métodos getString(…), getInt(…), getFloat(…), etc. antes de ejecutar la sentencia SQL.

--BIEN--

**Dado el siguiente código, y siendo “con“ un objeto de tipo Connection:**

***String sql = "DELETE FROM CLIENTE WHERE NIF=?";***

***String nif = "513332255L";***

***PreparedStatement sentencia = con.prepareStatement(sql); sentencia.setString(1, nif);***

***int valor = sentencia.executeUpdate();***



La variable valor contendrá true si la sentencia se ha ejecutado con éxito y false en caso contrario.



La variable valor contendrá el valor 1 si la sentencia SQL se ha ejecutado con éxito y 0 si ha ocurrido algún problema.



La variable valor contiene el número de filas que se han borrado al ejecutar la sentencia SQL.

--MAL—

Siria que es la subrayada

**Un objeto de la clase *ResultSet*...**



Sirve para ejecutar una sentencia SQL de tipo *SELECT*.



Representa una conexión con una base de datos que permite ejecutar una sentencia *SELECT*.



Se crea como resultado de la ejecución de una sentencia *SELECT* y contiene el conjunto de registros resultado de su ejecución.

El método executeQuery(…) de las clases Statement o PreparedStatement ejecuta una sentencia SQL de tipo SELECT y devuelve un objeto ResultSet con el conjunto de registros resultante.

--Bien--

**Dada la siguiente línea de código, y siendo sentencia un objeto de tipo *PreparedStatement*:**

***sentencia.setString(3, domicilio);***



Estamos sustituyendo un parámetro en la sentencia SQL y el 3 hace referencia al cuarto parámetro, ya que el primero es el 0.



Estamos sustituyendo un parámetro en la sentencia SQL y el 3 hace referencia al tercer parámetro.



Estamos sustituyendo el campo domicilio de la tabla por el valor de la tercera variable que esté definida en el programa Java.

--MAL--

**Cualquier tarea de manipulación de una base de datos con Java *JDBC* lleva consigo las siguientes tareas:**



1: establecer conexión con la base de datos. 2: interactuar con la base de datos. 3: cerrar la conexión con la base de datos.



1: cargar el driver en memoria. 2: establecer conexión con la base de datos. 3: interactuar con la base de datos. 4: cerrar la conexión con la base de datos.



1: cargar el driver en memoria. 2: establecer conexión con la base de datos. 3: interactuar con la base de datos.

Cualquier operación con bases de datos utilizando JDBC requiere de estas 4 tareas: 1. Cargar el driver en memoria. 2. Establecer conexión con la base de datos. 3. Interactuar con la base de datos. 4. Cerrar la conexión con la base de datos.

-BIEN—

**Indica cuáles de estas sentencias SQL pertenecen al *DML (Data Management Language*).**



*INSERT*



*DROP*



*DELETE*



*ALTER*



*UPDATE*



*CREATE*



*SELECT*

--BIEN—  
El DML está formado por el conjunto de sentencias SQL que tienen que ver con la manipulación de los datos (añadir registros, borrar registros, modificar registros, recuperar registros). Las sentencias que forman parte de este subconjunto son: INSERT, SELECT, UPDATE y DELETE.

**Dado el siguiente código SQL:**

***USE FERRETERIA;***

***BEGIN;***

***INSERT INTO CLIENTE VALUES***

***('12345678A','LUIS LOPEZ LOPEZ', 'C/ SOL, 3','639-639-639', 'MADRID');***

***INSERT INTO FACTURA VALUES(5446,'2017-12-15',0,'12345678A');***

***INSERT INTO DETALLE VALUES(5446,'MAR1',1,13);***

***INSERT INTO DETALLE VALUES(5446,'TOR7',2,0.9);***

***UPDATE PRODUCTO SET STOCK=STOCK-1 WHERE CODIGO='MAR1';***

***UPDATE PRODUCTO SET STOCK=STOCK-2 WHERE CODIGO='TOR7';***

***ROLLBACK;***

**¿Qué ocurre con la base de datos tras su ejecución?**



Que se ha añadido un cliente, una factura con sus líneas de detalle, y se ha actualizado el stock.



Aunque se ha añadido un cliente, una factura con sus líneas de detalle y se ha actualizado el stock, finalmente no ha ocurrido nada con la base de datos porque se han revertido los cambios con la sentencia ROLLBACK.



Se ha añadido una nueva factura, pero no se ha podido actualizar el stock.

--MAL--

**Indica cuáles de las siguientes palabras corresponden a estados en los que puede encontrarse una transacción.**



Activa.



Abierta.



Cerrada.



Parcialmente comprometida.



Fallida.



Cancelada.



Borrada.



Abortada.



Comprometida.

Una transacción puede encontrarse en uno de los siguientes estados: activa, parcialmente comprometida, fallida, abortada y comprometida.

--MAL—

**Dada la siguiente cadena de conexión:**

***String cadenaConexion = "jdbc:mysql://Perico:3308/Mercado”;***

**Marca las afirmaciones correctas.**



*Jdbc:mysql* es el proveedor.



*Mercado* es el nombre del servidor.



*3308* es la contraseña del usuario con permisos para uso de la base de datos.



*Perico* es el nombre del servidor.



*Mercado* es el nombre de la base de datos.



*Perico* es el nombre de la base de datos.



*3308* es el puerto.

La cadena de conexión especificada en esta pregunta tiene el siguiente formato: proveedor://servidor:puerto/Nombre\_BD.

-MAL--

**Dada la siguiente sentencia:**

***con = DriverManager.getConnection(cadenaConexion, a, b);***



La variable *a* debería contener la contraseña de acceso a la base de datos.



La variable *a* debería contener el nombre de usuario que tiene permiso para acceder a la base de datos.



La variable *a* debería contener el nombre del DBMS.

El método getConnection tiene la siguiente estructura: getConnection(cadena\_conexión, usuario, contraseña);

-BIEN--

**Una transacción es...**



Un listado completo de una tabla de la base de datos.



Una consulta de selección que implica la consulta en varias tablas de la base de datos.



Un conjunto de operaciones de actualización de la base de datos que deben realizarse todas o ninguna.

Una transacción es el conjunto de operaciones necesarias para llevar a cabo una determinada tarea que implique modificaciones en la base de datos. Dichas operaciones deben tratarse como una única unidad de trabajo, de modo que se efectúen todas o ninguna.

--BIEN--

**El dominio de un campo o atributo es:**



El entorno que delimita dónde puede utilizarse dicho campo.



El conjunto de valores posibles que puede tomar dicho campo.



La tabla donde está dicho campo.

--MAL—

**Cuando trabajamos con *JDBC* utilizamos *Class.forName* para:**



Cargar en memoria el *JDBC*.



Cargar en memoria todas las librerías de *java.sql.*



Cargar en memoria los drivers.

-MAL--

***DriverManager*...**



Actúa como proveedor de objetos Connection siempre y cuando hayamos cargado en memoria el driver que corresponda con el proveedor de base de datos requerido.



Actúa como proveedor de objetos Connection y tiene incorporados muchos drivers de forma predeterminada.



Representa una conexión con una base de datos.

DriverManager requiere que se hayan cargado en memoria previamente los drivers de los proveedores de base de datos con los que vayamos a trabajar.

--BIEN—

**Indica cuáles de estos atributos serían idóneos como clave principal:**



*Nombre*



*Telefono*



*Edad*



*NIF*



*CodigoArticulo*



*NumeroFactura*



*FechaFactura*

--BIEN--

La clave principal no puede tener, en ningún caso, valores duplicados. El atributo Nombre no valdría, porque es posible que existan dos personas con el mismo nombre. Los atributos Telefono, Edad y FechaFactura, sin embargo, podrían tener valores duplicados.